ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ»
(Н И У « Б е л Г У »)

ИНСТИТУТ ИНЖЕНЕРНЫХ И ЦИФРОВЫХ ТЕХНОЛОГИЙ

КАФЕДРА ИНФОРМАЦИОННЫХ И РОБОТОТЕХНИЧЕСКИХ СИСТЕМ

СИСТЕМА «УМНОЙ» ВЕНТИЛЯЦИИ

Выпускная квалификационная работа

обучающегося по направлению подготовки
09.03.02. Информационные системы и технологии
очной формы обучения,
группы 12001509
Адамович Владислава Андреевича

Научный руководитель
доцент к.ф.-м.н. Мигаль Л.В.

БЕЛГОРОД 2019

Содержание

ВВЕДЕНИЕ

Климатическая техника прочно вошла в нашу повседневную жизнь. Системы отопления, вентиляции и кондиционирования есть практически в каждом современном доме. Кроме этого, при необходимости можно установить увлажнители, очистители и другие приборы, которые создают благоприятный микроклимат в помещении.

Оптимальные параметры климата помещения постоянно меняются, это значит, что использование ЭВМ для автоматизации управления различными системами является приоритетным вариантом решения. Перечисленные устройства, имеющие подобное решение, требуют контроля и управления, и их можно отнести к технологии Интернета вещей, как части «умного» дома.

В данной работе будет показан процесс проектирования системы «умной» вентиляции «HouseCloud», которое должно иметь следующе возможности:

– возможность удаленного управления вентиляцией в помещении;

– возможность отображения текущего состояния климата в помещении;

– возможность автоматического контроля климатом в помещении.

Также разрабатываемая система будет иметь более низкую стоимость в сравнении с аналогичными системами.

Целью выполнения данной работы является усовершенствование процесса управления устройствами вентиляции.

Для достижения поставленной цели были сформулированы следующие задачи:

– анализ предметной области;

– выбор необходимых компонентов и технологий;

– проектирование прототипа устройства управления «умной» вентиляции;

– разработка программного обеспечения для устройства управления «умной» вентиляции;

– разработка интерфейса мобильного приложения «HouseCloud»;

– разработка бэкенда мобильного приложения «HouseCloud»;

– тестирование работы «умной» вентиляции.

Выполнение поставленных задач позволит удаленно управлять параметрами климата в помещении.

1 Анализ рынка устройств управления вентиляцией и выбор инструментов проектирования

1.1 Анализ систем «умной» вентиляции

У ведущих мировых компаний, занимающихся развитием IT-технологий в области «умного» дома процессы автоматизации управления климатом в списке решаемых задач стоят на втором месте (после автоматизации работы с осветительными приборами). Спектр применяемого оборудования в данном сценарии очень широк. Верхние строки занимает профессиональные устройства для HVAC (Heating Ventilation & Air Conditioning – отопление вентиляция и кондиционирование) [1], включая теплые полы, системы рециркуляции воздуха и холодные потолки. В доступном сегменте можно увидеть традиционные бытовые кондиционеры и многочисленные электронагревательные приборы и газовые котлы, а задачи вентиляции чаще всего решаются «ручным» методом открытия окон.

Решением вопросов управления вентиляцией в современном доме является подключение к ней по специальному интерфейсу определенного устройства, обеспечивающего не только отправку команд, но и контроль статуса, а также информирование о состоянии и возможных неисправностях. Однако этот вариант доступен только в определенных моделях, может требовать приобретение дополнительных элементов.

Существующие системы используют следующие стандарты:

- AllJoyn – это стандарт с открытым исходным кодом, предназначенный для взаимодействия приложений, устройств и пользователей через WiFi и Bluetooth (и другие беспроводные технологии передачи данных) вне зависимости от типа устройства.

- HomeKit – это стандарт от Apple, который позволяет пользователям использовать устройство iOS для настройки, общения и управления умными домашними устройствами.

- SmartThings – это стандарт для домашней автоматизации, которые имеют возможность контроля и отслеживания деятельности своих совместимых устройств с помощью единого универсального приложения для удаленного управления.

- Z-Wave – это стандарт, который является беспроводным протоколом связи, разработанным для домашней автоматизации, в части контроля и управления в жилых и коммерческих объектах. Технология использует маломощные и миниатюрные радиочастотные модули, которые встраиваются в бытовую электронику и различные устройства, такие как осветительные приборы, приборы отопления, устройства контроля доступа, развлекательные системы и бытовая техника.

Подобные стандарты начинают появляться в более массовом сегменте, но имеют большую себестоимость при внедрении.

Стоимость подобных решений можно уменьшить, разработав отдельное устройство, которое смогло бы управлять уже имеющимися устройствами вентиляции. То есть, устройство будет управлять кондиционерами, которые не поддерживают новые технологии управления.

1.2 Анализ существующих разработок и обоснование выбора средств проектирования

На данный момент устройств «умного» управления вентиляцией не так много на рынке. А те, что есть, имеют свои ограничения и высокую цену.

Наиболее распространенные приведены в таблице 1, в которой показана общая сравнительная характеристика. Если проанализировать эти данные, то можно выделить особенность, которая есть у представленных устройств – это

высокая стоимость, которая может отрицательно влиять на популярность и продажи этих устройств.

Таблица 1 – Сравнительная характеристика устройств управления

Наименование, краткое описание	Внешний вид устройства	Особенности устройства	Стоимость
УРК-2Т Позволяет чередовать работу кондиционеров, с указанным периодом времени		ИК управление, питание от сети, GSM.	10026 руб.
Philio Управляет кондиционерами посредством ИК команд переводит команды полученные от Z-Wave контроллер на язык, понятный кондиционеру.		3*AAA Батарейки, Z-Wave контроллер, ИК управление	8400 руб.
ZXT-120 Управляет кондиционерами посредством ИК команд переводит команды полученные от Z-Wave контроллер на язык, понятный кондиционеру.		3*AAA Батарейки, Z-Wave контроллер, ИК управление	8820 руб.

Можно также заметить, что устройства предлагаются с дополнительными модулями, которые продаются отдельно, за дополнительную стоимость. Минусом таких устройств является, что они управляются при помощи отдельного модуля. В УРК-2Т модуль управления встроен в корпус устройства. Philio – управляется отдельными кнопками и регуляторами. ZXT-120 – управление через отдельный модуль. Еще одним минусом можно назвать минимальный набор настроек управления.

1.3 Постановка задач

Для достижения поставленной цели необходимо выделить задачи, которые будут решаться по мере выполнения данной работы.

Необходимо спроектировать прототип устройства управления вентиляцией, которое позволит удаленно управлять соответствующим оборудованием в помещении и передавать текущую информацию о параметрах состояния, что является промежуточным звеном между устройствами вентиляции и мобильным приложением.

Для управления данным устройством необходимо разработать программу, которая будет получать данные с датчиков, находящихся в устройстве управления, обрабатывать команды пользователя, которые поступают с мобильного приложения и передавать текущие данные пользователю на смартфон.

Далее необходимо разработать интерфейс для мобильного приложения. Он должен соответствовать современным подходам разработки интерфейса, поскольку обычному пользователю проще понять уже привычный интерфейс.

Также необходимо разработать программное обеспечение, которое позволит достичь поставленных функциональных задач системы управления «умной» вентиляцией.

Выводы по первому разделу

В данном разделе был проведен анализ предметной области и проанализирован рынок устройств систем «умной» вентиляции. Обоснованы поставленные задачи исследования.

2 Обоснование использования ЭВМ и выбор технологий для проектирования устройства «умной» вентиляции «HouseCloud»

2.1 Обоснование необходимости использования вычислительной техники для решения задач разработки устройства

Интеграция информационных технологий и концепций, активно развивающихся в XXI веке, создает предпосылки к формированию локальных и даже национальных киберфизических систем, которые в свою очередь формируют технологии Интернета вещей, и как следствие, являются частью систем «умного» дома.

Киберфизические системы (Cyber-Physical System, CPS) – это системы, состоящие из различных природных объектов, искусственных подсистем и управляющих контроллеров, позволяющих представить их как единое целое. В CPS обеспечивается тесная связь и координация между вычислительными и физическими ресурсами. Компьютеры осуществляют мониторинг и управление физическими процессами с использованием такой обратной связи, где происходящее в физических системах оказывает влияние на вычисления и наоборот.

Немецкая академия Acatech уже говорит о перспективах национальных киберфизических платформ, которые складываются из трех типов сетей:

– Интернет людей;

– Интернет вещей;

– Интернет сервисов.

Схема взаимодействия «умных» систем представлена на рисунке 1. На это схеме можно увидеть насколько широко распространение «умных» систем в современном мире.

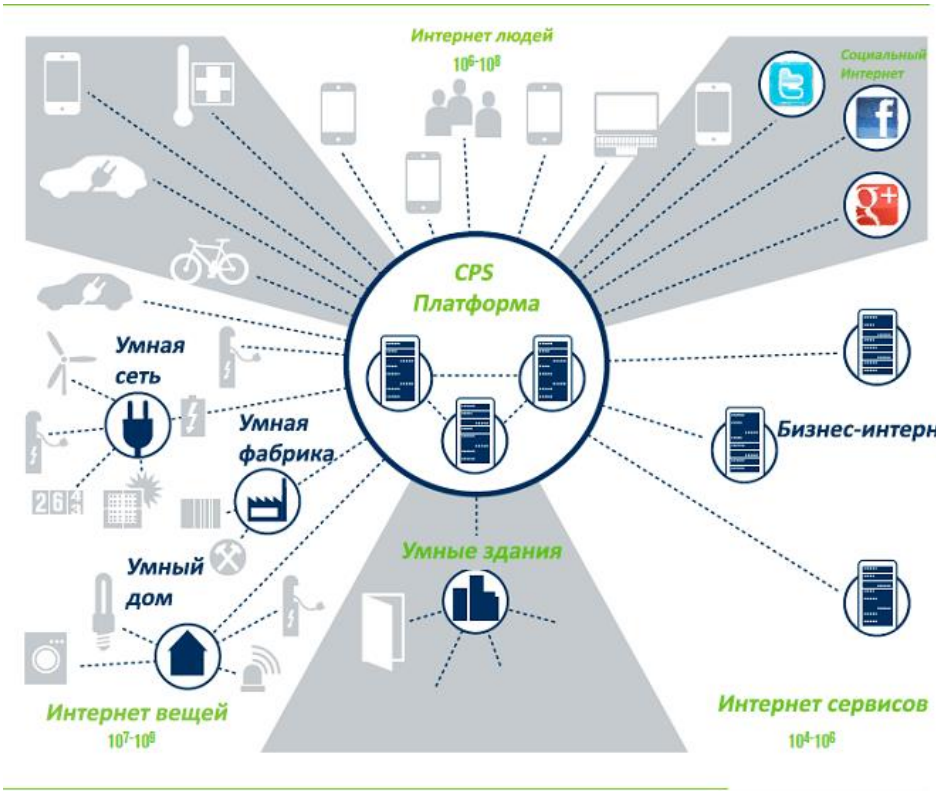


Рисунок 1 – Схема взаимодействия «умных» систем

Согласно статистике (по данным компании Hootsuite), представленной на рисунке 2 можно заметить, что на 2018 год 5 млрд. человек имеют мобильные телефоны, в том числе и смартфоны. Такое количество объясняется тем, что в некоторых случаях среднестатический пользователь имеет несколько мобильных устройств, например, умные часы, планшет.

Рисунок 2 – Статистические данные количества пользователей

По данным компании Hootsuite, с января 2017 года российских пользователей в Интернет стало больше на 5 миллионов, что составляет увеличение на 4%, а социальными сетями теперь пользуются на 9 миллионов больше людей (+ 15% по сравнению с данными 2016 г.) [2].

Уникальные пользователи мобильных устройств составляют 5,1 млрд. человек, что составляет 68%. Данный процент повышается с каждым годом, так как количество используемых устройств растет.

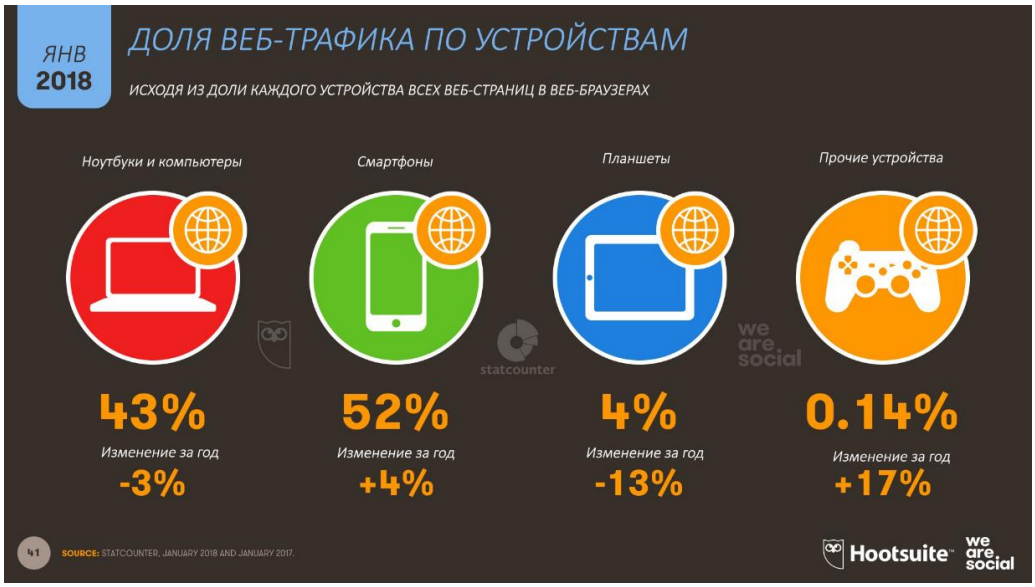


Рисунок 3 – Статистические данные веб-трафика устройств

На рисунке 3 показан срез показателей веб-трафика по устройствам. Большую часть составляют мобильные устройства, далее идут персональные компьютеры, оставшуюся часть занимают планшеты и прочие устройства.

Также можно заметить, что процент роста показателей говорит о том, что люди все меньше пользуются компьютерами, чаще используют смартфоны и другие альтернативные устройства.

2.2 Обоснование выбора средств проектирования

Учитывая недостатки существующих устройств, в данной работе предполагается использовать единый модуль, который будет управлять всеми устройствами в сети – смартфон. Отметим, что стоимость производства мобильного приложения будет гораздо ниже себестоимости разработки и

производства отдельного модуля управления устройством «умной» вентиляции.

В данный момент имеются две основные мобильные операционные системы (ОС): IOS – производства купертиновской компании Apple; Android – производства компании Alphabet Inc., также известной как Google.

В данной работе будет разработано мобильное приложение на ОС Android, поскольку проприетарные условия разработки на платформе IOS, требуют больших начальных вложений, таких как устройство с ОС MacOS, так и статуса разработчика, который имеет стоимость 99$, для производства на платформе Android требуется лишь статус разработчика, стоимость которого составляет 25$.

Для разработки мобильного приложения на выбранной ОС, имеется три основных языка программирования.

Первым рассмотрим Xamarin – это фреймворк для кроссплатформенной разработки мобильных приложений (iOS, Android, Windows Phone) с использованием языка C#, разработанный компанией Microsoft.

Данный язык отличается от других, тем что приложение одновременно разрабатывается сразу на 3 платформах. Но имеет минусы. Большой объем готовых приложений. Как следствие большого объема приложений низкая производительность. А эти факторы основополагающие для разработки мобильных приложений.

Далее рассмотрим язык Kotlin – это статически типизированный язык программирования, работающий поверх JVM, разработанный компанией JetBrains.

Данный язык появился достаточно недавно и уже имеет свою аудиторию пользователей. Но так как этот язык появился недавно, он мало распространён и как следствие мало протестирован на всевозможные ошибки и уязвимости.

Java – объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems. Наиболее устоявшийся язык программирования среди мобильных приложений. Приложения, разработанные на Java, имеют небольшой объем, скорость обработки выше аналогов, а также

имеются гибкие возможности разработки интерфейса программ. Интерфейс будет разработан при помощи языка разметки XAML. XAML – это язык разметки, который появился вместе с первой версией WPF от Microsoft.

Далее рассмотрим платы, которые используются для разработки систем «умного» дома. Основные платы, которые имеются на рынке:

– Raspberry Pi – одноплатный компьютер, изначально разработанный как бюджетная система для обучения информатике, но позже получивший более широкое применение в системах умного дома, медийных систем и небольших компьютерах. Разрабатывается компанией Raspberry Pi Foundation. За 5 лет было продано более 12,5 миллионов устройств Raspberry Pi. Поддерживает большинство ОС, таких как: Windows 10 IOT и большинства дистрибутивов Linux. Стоимость в диапазоне от 2000 до 5000 рублей.

– Intel NUC – это производительный мини-ПК от компании Intel. Имеет размер 10x10 см и используется для развлечений, игр и эффективной работы. Он поддерживает индивидуальное конфигурирование платы с возможностью выбора памяти, подсистемы хранения и операционной системы. Стоимость начинается от 8000 рублей и достигает до 60000 рублей, в зависимости от комплектующих.

– Arduino – микроконтроллер на базе ATmega328, логическая микросхема для обработки данных с тактовой частотой 16 МГц, имеющая 8 аналоговых и 14 цифровых контактов общего назначения. Стоимость в диапазоне от 300 рублей до 900 рублей, в зависимости от форм фактора платы.

Исходя приведенных выше плат для построения «умного» дома наиболее подходящим для данной работы является Arduino. Поскольку мощностей Intel NUC не требуется, а стоимость Raspberry Pi высокая.

Далее необходимо подобрать датчики и компоненты для разработки устройства. Для построения такого устройства, необходимы следующие компоненты:

– Плата WiFi (Wireless Fidelity – беспроводная привязанность, беспроводной стандарт передачи данных на частоте от 2402 до 2480 МГц).

– Датчик температуры.

– Датчик влажности.

– ИК передатчик (ИК – инфракрасный передатчик).

– Дисплей.

Далее приведены модели и характеристики данных компонент.

Дисплей – LCD 1602 с интерфейсом I2C, представляет из себя жидкокристаллический экран, невысокая стоимость, имеются различные модификации с разными цветами подсветки. Он показан ниже на рисунке 4.

Рисунок 4 – Жидкокристаллический 1602 экран «WAVGAT»

Ниже приведены технические характеристики модуля:

– символьный тип отображения, есть возможность загрузки символов;

– светодиодная подсветка;

– контроллер HD44780;

– напряжение питания 5В;

– формат 16x2 символов;

– диапазон рабочих температур от -20С до +70С, диапазон температур хранения от -30С до +80 С.

Сенсор температуры и влажности располагается на одной плате DHT11, как показано на рисунке 5. Он позволяет измерять температуру в диапазоне от 0 до 50 градусов с точностью 2 градуса, определение влажности от 20% до 95% с 5% точностью [4]. Также ниже приведены технические характеристики модуля:

– потребляемый ток – 2,5 мА (максимальное значение при преобразовании данных);

– измерение влажности в диапазоне от 20% до 80%. Погрешность может составлять до 5%;

– габаритные размеры: 15,5x12x5,5 мм;

– питание – от 3 до 5 Вольт;

– одно измерение в единицу времени (секунду), частота составляет 1 Гц.

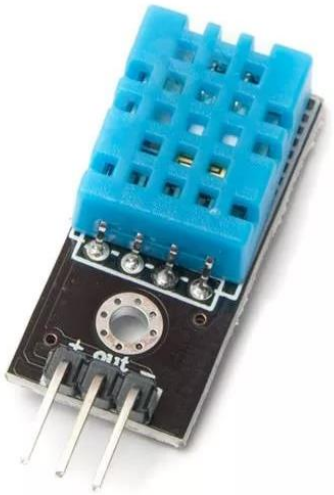


Рисунок 5 – Сенсор температуры и влажности DTH11

Модуль WiFI ESP8266-01 показан на рисунке 6. Процессор Tensilica имеет 80 MHz 32-bit. Поддерживает стандарты передачи данных IEEE 802.11 b/g/n Wi-Fi с WEP и WPA/WPA2.То есть может работать с любыми современными роутерами WiFi, которые используются в помещениях. Питание от 2,2 до 3,6 В. Интегрированный стек TCP/IP, HTML.

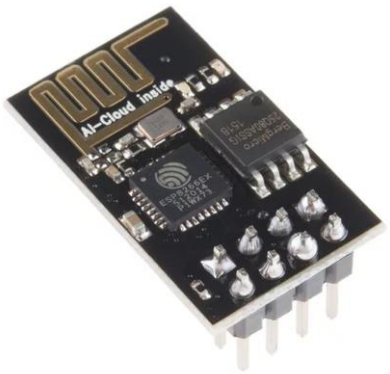


Рисунок 6 – Модуль WiFI ESP8266-01

ИК-передатчик, позволяющий передавать команды на вентиляционные устройства представлен на рисунке 7. Он основан на передачи светового луча в инфракрасном диапазоне на частоте 38kHz и длиной волны 940нМ. На плате установлен инфракрасный светодиод типа «TSAL6200». Интерфейс соединения – цифровой.

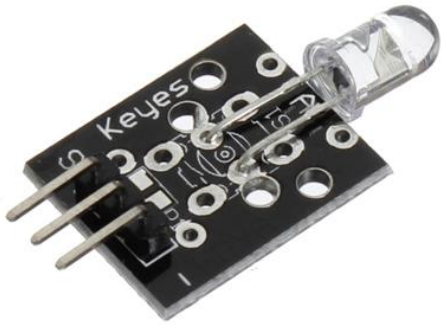


Рисунок 7 – ИК передатчик «Vishay»

Данные компоненты необходимы для разработки прототипа устройства управления «умной» вентиляции.

Также существует необходимость в разработке серверного приложения для обработки данных с устройства управления и мобильного приложения.

Наиболее распространенные языки программирования для разработки серверных приложений:

– PHP – скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений;

– NodeJS – это опенсорсная кроссплатформенная среда выполнения для JavaScript, которая работает на серверах;

– Asp. NET – платформа разработки веб-приложений, в состав которой входит: веб-сервисы, программная инфраструктура, модель программирования (от компании Microsoft).

Для разработки серверного приложения был выбран язык NodeJS. Поскольку он наиболее похож по семантике на Java, а также позволяет развертывать системы различных масштабов.

Далее необходимо выбрать систему управления базой данных (СУБД) из приведенного ниже списка:

– PostgreSQL – свободная объектно-реляционная система управления базами данных. Существует в реализациях для множества UNIX-подобных платформ, включая AIX, различные BSD-системы, HP-UX, IRIX, Linux, macOS.

– MySQL – свободная реляционная система управления базами данных. Разработку и поддержку MySQL осуществляет корпорация Oracle, получившая права на торговую марку вместе с поглощённой Sun Microsystems.

– Oracle Database – объектно-реляционная система управления базами данных компании Oracle.

Для данной работы будет использоваться СУБД MySQL, поскольку она достаточно безопасна и устойчива к нагрузкам.

Данные средства проектирования позволят построить систему «умной» вентиляции, которая состоит из устройства управления «умной» вентиляции, мобильного и серверного приложений.

Выводы по второму разделу

В данном разделе был обоснован выбор комплектующих и программных инструментов для решения поставленных задач. Для разработки данного устройства было принято решение использовать: плату Arduino Uno, язык высокого уровня Java, язык программирования для серверных приложений NodeJS, СУБД MySql.

3 Проектирование системы управления вентиляцией

3.1 Разработка схемы функционирования системы «HouseCloud»

Разрабатываемое мобильное приложение будет получать данные с датчиков устройства, и передавать команды для управления устройствами вентиляции. На рисунке 8 показана схема взаимодействия узлов.

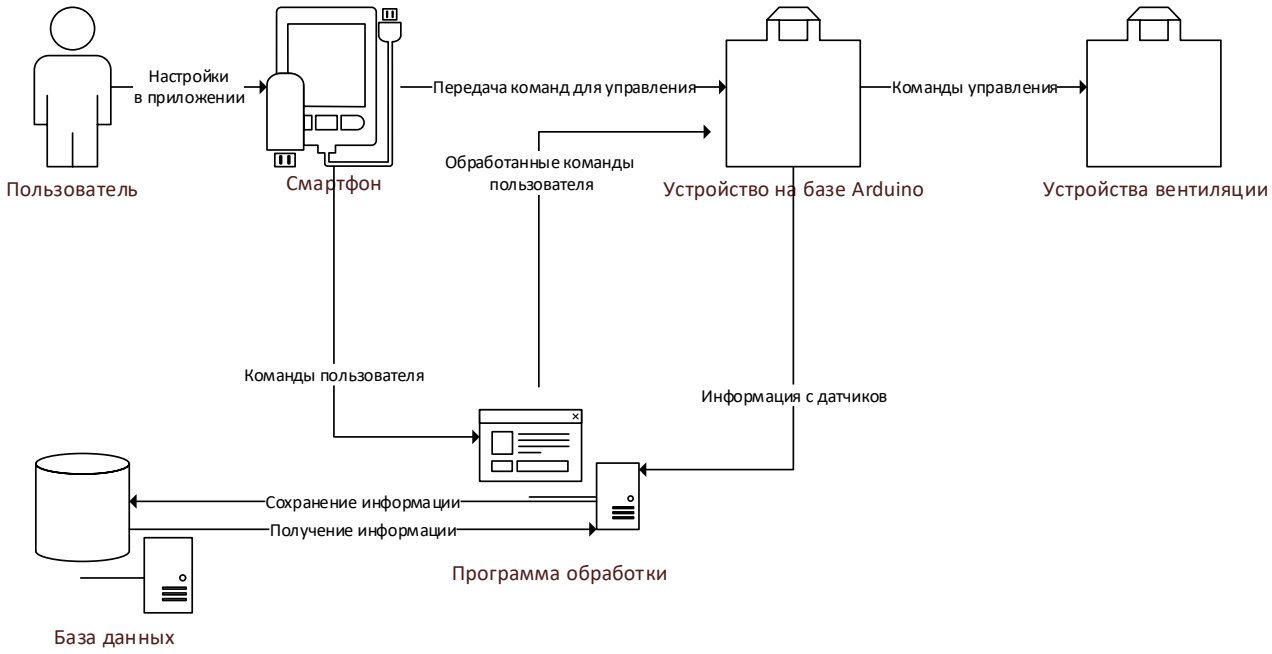


Рисунок 8 – Общая схемы работы системы «HouseCloud»

В полученной системе можно выделить несколько основных узлов:

– устройство на базе Arduino;

– программа обработки;

– мобильное приложение;

– устройство вентиляции.

Мобильное приложение позволит управлять устройствами, а также наглядно отображать текущие данные работы системы. Будет возможность настройки температуры и дополнительных универсальных функций для автоматического регулирования соответствующими параметрами. Также будет возможность настройки для подключения к устройству управления.

Программа обработки будет реализована в виде отдельного серверного приложения, которое будет выполнять роль регулятора и обработчика данных. Для этого необходимо подключение к сети Интернет.

Устройствами вентиляции данной системы являются кондиционеры, системы увлажнения воздуха, ионизаторы и устройства очистки воздуха.

Данное взаимодействие позволит достичь поставленной цели.

3.2 Разработка устройства управления «умной» вентиляцией

Выбранные компоненты позволяют разработать прототип устройства управления «умной» вентиляцией. Далее будет показано подключение и соединение всех компонент в единую схему.

На рисунке 9 показан модуль WiFi ESP8266-01 подключенный к контроллеру Arduino Uno.

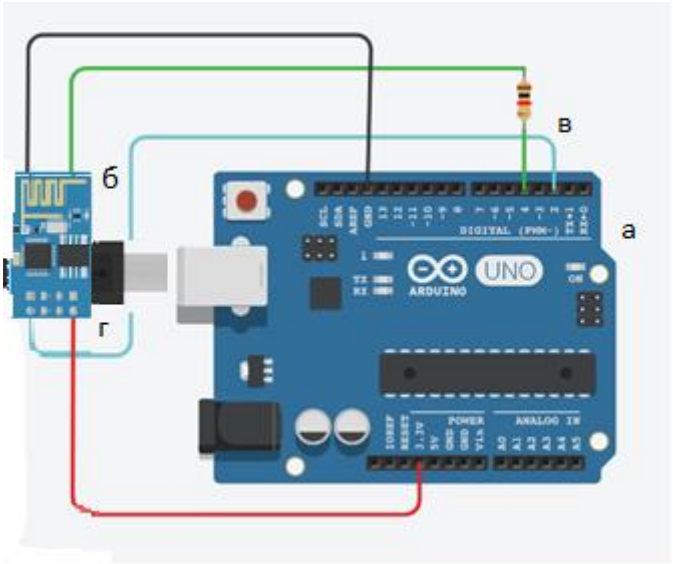


Рисунок 9 – Подключение модуля WiFi (а – плата Arduino;
б – WiFi ESP8266-01; в – резистор 220 ОМ; г – источник питания)

Плата расширения модуля WiFi имеет 4 пина: VCC, GND, TX, RX. Можно подключить как к цифровым портам, так и к портам TX и RX на плате Arduino. Далее приведены расшифровки контактов платы:

– VCC – плюс питания 3.3 В;

– GND – минус питания 3.3 В;

– RX – выход для передачи данных;

– TX – вход для приема данных;

– RST – кратковременно подав на неё низкий логический уровень, модуль перезагрузиться. Лучше прижать ее через резистор 10 кОм к плюсу питания;

– CH_PD – служит для перевода модуля в энергосберегающий режим, в котором он потребляет очень маленький ток. Ну и снова – не будет лишним «прижать» эту ногу резистором на 10 кОм к плюсу питания.

WiFi позволит отправлять и получать данные через сеть Интернет, для удаленного управления устройствами вентиляции.

Ниже представлен фрагмент кода, который реализует работу данного компонента:

```
SoftwareSerial wifiSerial(2, 3);      // RX, TX for ESP8266
bool DEBUG = true;
int responseTime = 10;
void setup()
{
  pinMode(13,OUTPUT);
    Serial.begin(115200);
  wifiSerial.begin(115200);
  while (!wifiSerial) {
    ;
  }
  sendToWifi("AT+CWMODE=2",responseTime,DEBUG);
sendToWifi("AT+CIFSR",responseTime,DEBUG);
  sendToWifi("AT+CIPMUX=1",responseTime,DEBUG);
sendToWifi("AT+CIPSERVER=1,80",responseTime,DEBUG);
  sendToUno("Wifi connection is running!",responseTime,DEBUG);
        }
```

На рисунке 10 представлена схема подключения датчика температуры к плате Arduino UNO.

Как можно увидеть на рисунке имеется 3 пина для подключения: VCC, GND, Data, а также подключение пина для передачи данных к цифровому порту платы.

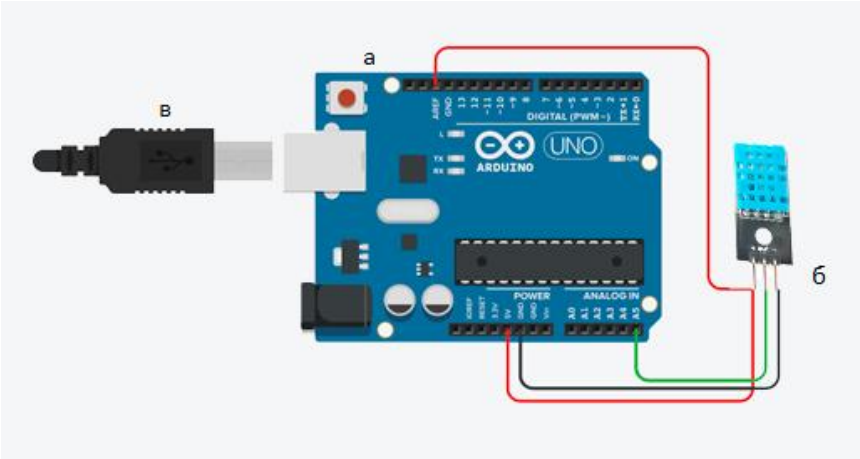


Рисунок 10 – Подключенный датчик температуры и влажности (а – плата Arduino; б – датчик температуры и влажности DHT11; в – источник питания)

Ниже представлен фрагмент кода, который реализует работу данного датчика:

```
tempReading = analogRead(tempPin);
  delay(100);
Serial.print("Temp reading = ");
Serial.print(tempReading);
float voltage = tempReading * aref_voltage ;
voltage /= 1024.0;
temperatureC = (voltage - 0.5) * 100 ;
Serial.print(temperatureC);
Serial.println(" degrees C");
```

Здесь используется аналоговый вход, данные которого необходимо преобразовывать к градусам Цельсия.

После расчетов данные выводятся в строку логов.

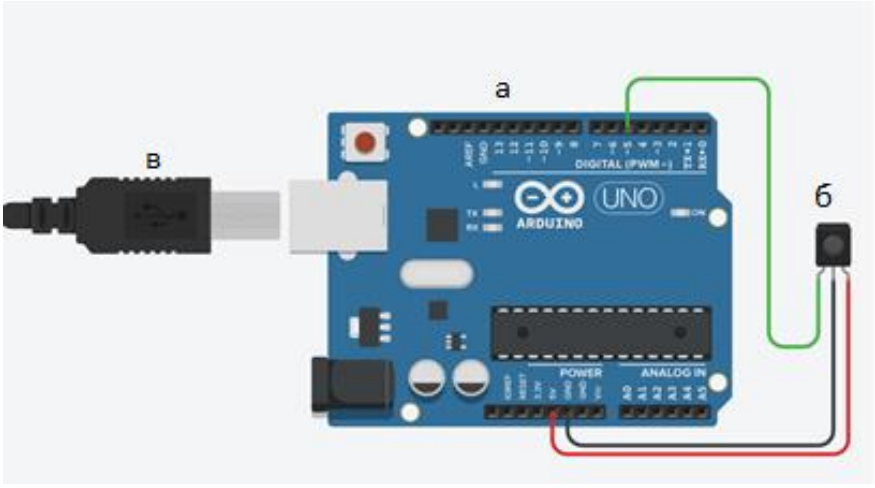


Рисунок 11 – Подключение модуля ИК-передатчика (а – плата Arduino; б – ИК передатчик; в – источник питания)

На рисунке 11 показан модуль ИК передатчика. Имеет 3 пина: VCC, GND, DATA. Подключение аналогично датчику температуры и важности.

Использование данного модуля предоставит возможность управлять устройствами вентиляции через ИК-порт.

Ниже представлен код, в нем указаны команды, которые будут передаваться на кондиционеры:

```
iarduino_IR_TX VD(2);
void setup(){
  pinMode(3,INPUT);
  pinMode(4,INPUT);
  VD.begin();
}
void loop(){
  if(digitalRead(3)){
      VD.send(0x00000000000007F7, true);
   }
    if(digitalRead(4)){
       VD.send(0x0000000000000017 true);
   }
}
```

Коды представляют в виде трехзначных или четырехзначных чисел, которые отправляются в виде 16-битного шестнадцатеричного кода, в дальнейшем они будут взяты у производителей.

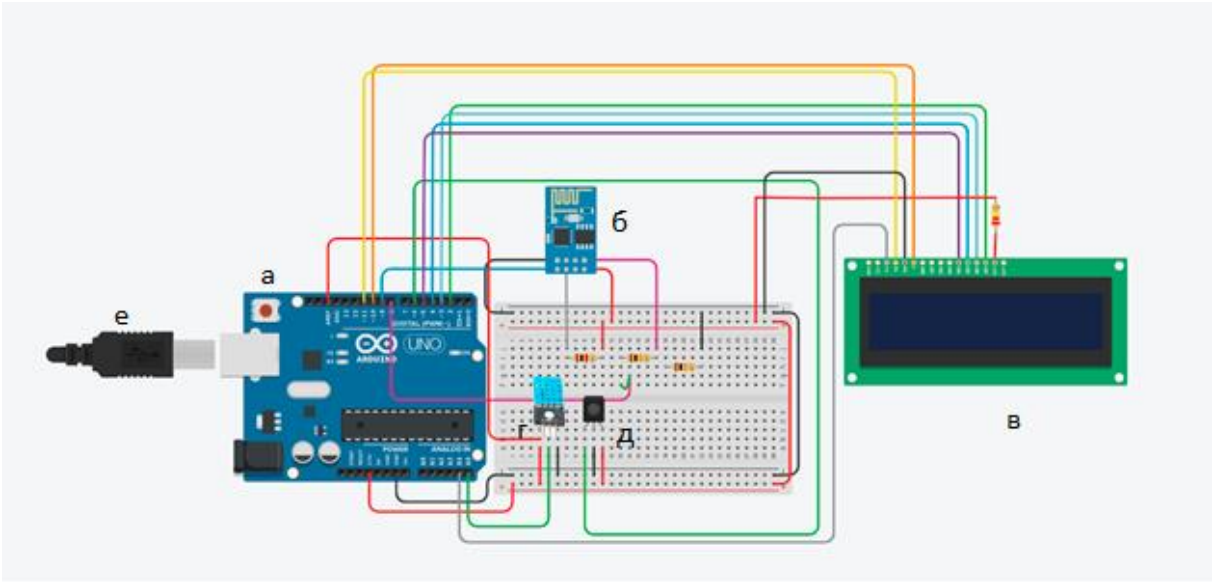


Рисунок 12 – Прототип устройства «HouseCloud» (а – плата Arduino; б – WiFi ESP8266-01; в – жидкокристаллический экран 1602; г – датчик DHT11; д – инфракрасный передатчик; е – источник питания)

На рисунке 12 показана собранная схема прототипа устройства «HouseCloud».

Далее будет описан алгоритм работы разрабатываемого устройства.

На рисунке 13 приведена блок-схема работы программы данного устройства.

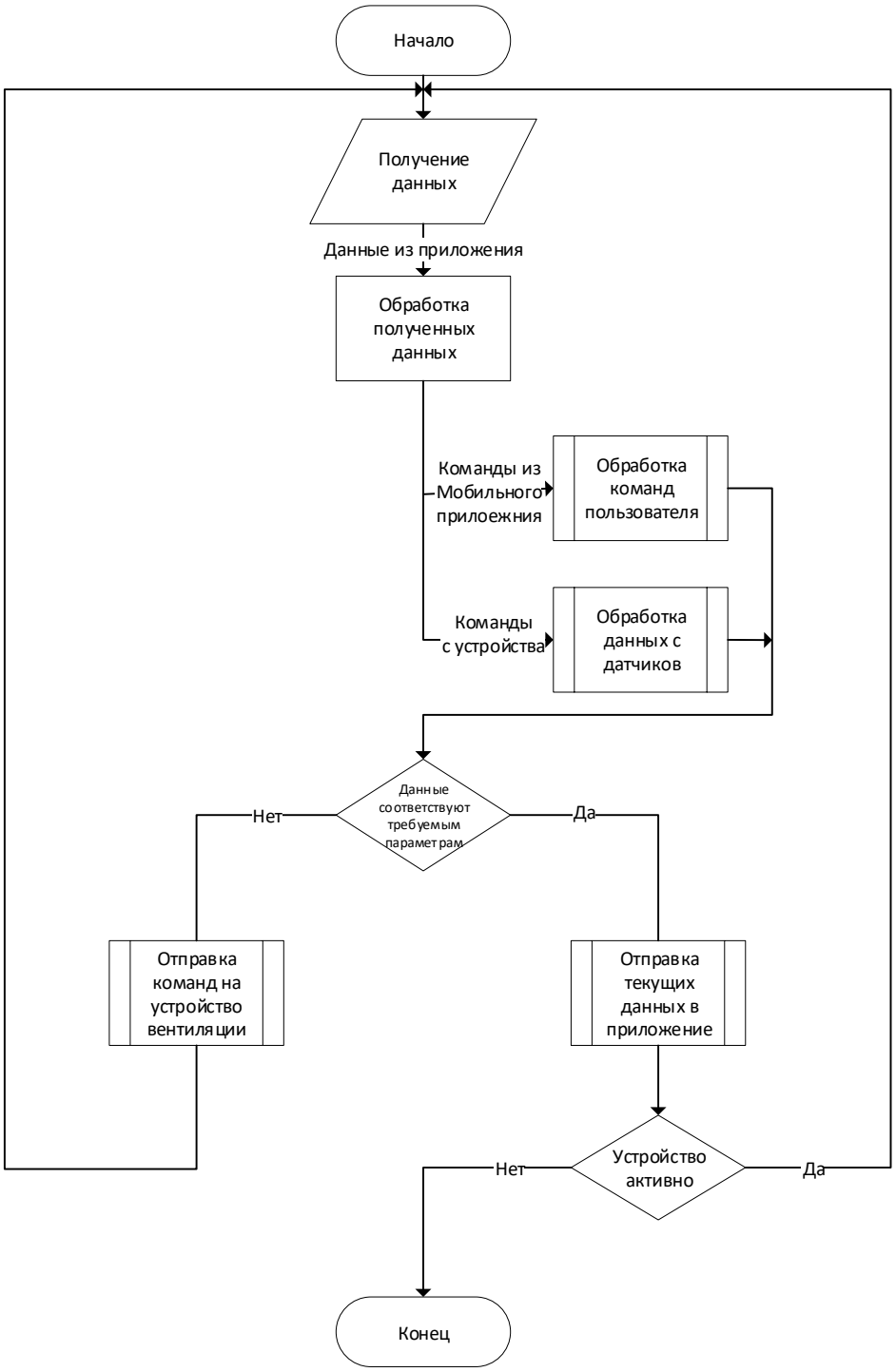


Рисунок 13 – Блок-схема работы устройства «HouseCloud»

Алгоритм работы устройства начинается с получения данных. После этого устройство обрабатывает полученные данные, и исходя из них программа распределяет либо данные получены с датчиков, либо с мобильного устройства. И на основании этой информации определяет ход действий.

После этого устройство оценивает данные на соответствие введенным параметрам. Если температура соответствует, то система отправляет оповещение о достижении установленной температуры в мобильное приложение. Или, если не соответствует, то отправляет команды на устройство вентиляции, после чего оно начинает свою работу. После этого система проходит цикл работы, пока условие не будет выполнено, либо устройство не будет выключено.

3.3 Разработка интерфейса приложения

Проектирование интерфейса было разделено на 2 этапа. Первый – это проектирование дизайна в графическом редакторе (Adobe Photoshop). Второй – это перенос дизайна в продуктивный проект, на соответствующий язык XAML. Такой подход позволяет разделить работу дизайнера и программиста [5].

Ниже приведена часть кода XAML, в которой показана код вертикального макета для формирования свойств и параметров для выстраивания других макетов, которые содержат компоненты экрана формы приложения:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android=http://schemas.android.com/apk/res/android

    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools=http://schemas.android.com/tools

    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:weightSum="3"

    tools:context=".MainActivity"
    tools:showIn="@layout/app_bar_main">
```

На рисунке 14 представлена форма главного меню «HouseCloud».

Рисунок 14 – Главное меню «HouseCloud»

Далее представлен код, который формирует горизонтальный макет для отображения компонент экрана:

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_weight="1"
    android:orientation="horizontal"
    android:weightSum="2">
```

Далее представлен код, который формирует компонент «Кнопка». Имеется картинка фона, а также обработчик события:

```
    <Button
        android:id="@+id/gostin"
        style="@android:style/Widget.DeviceDefault.Button.Borderless"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:background="@drawable/zal"
                android:onClick="onClickgostin" />
```

Далее была разработана форма, в которой настраивается температура и применяютя другие функции вентиляции. Прототип интерфейса показан на рисунке 15.

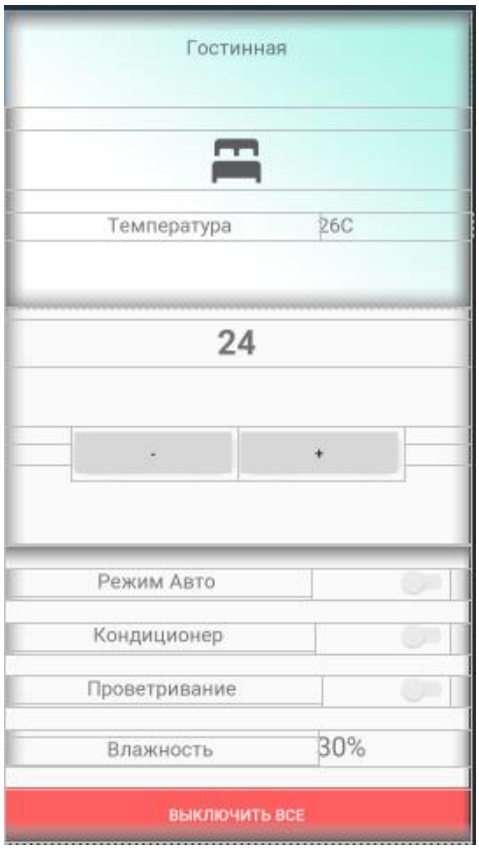


Рисунок 15 – Окно управления температурой в гостиной

Далее показан код XAML, в котором представлен листинг вертикального макета формирования свойств и параметров для выстраивания других макетов, которые содержат компоненты экрана формы приложения:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
          tools:context=".spaln">
```

Далее представлен код, который формирует горизонтальный макет для отображения компонент экрана:

```
<LinearLayout
```

```
android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_weight="3"
android:background="@drawable/wall2"
android:orientation="horizontal">
```

Далее представлен код, который формирует компонент текст:

```
TextView
        android:id="@+id/textView6"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="4"
        android:gravity="center_horizontal"
        android:paddingTop="20dp"
        android:text="@string/spaln"
        android:textSize="18sp" />
```

Далее представлен код, который формирует текст с переключателем в оболочке макета.

```
<LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginTop="20dp"
        android:layout_weight="1"
        android:fadingEdge="horizontal|vertical"
        android:orientation="horizontal"
        android:weightSum="3">

        <TextView
           android:id="@+id/text_auto_gostin"
           android:layout_width="wrap_content"
           android:layout_height="wrap_content"
           android:layout_weight="2"
           android:gravity="center"
           android:text="@string/auto_set"
           android:textSize="18sp" />

        <Switch
           android:id="@+id/switch_auto_gostin"
           android:layout_width="wrap_content"
           android:layout_height="wrap_content"
           android:layout_marginRight="20dp"
           android:layout_weight="1" />
              </LinearLayout>
```

Таким образом, был разработан интерфейс мобильного приложения «HouseCloud» с помощью языка XAML.

3.4 Разработка бэкенда приложения «HouseCloud»

Разработка логики приложения была начата с функционального проектирования работы приложения. Был выбран стандарт IDEF0, так как он достаточно универсален для описания работы системы [6].

Разработка схемы была проведена в программе AllFusion Process Modeler r7. На рисунке 16 показана контекстная диаграмма «КАК БУДЕТ».

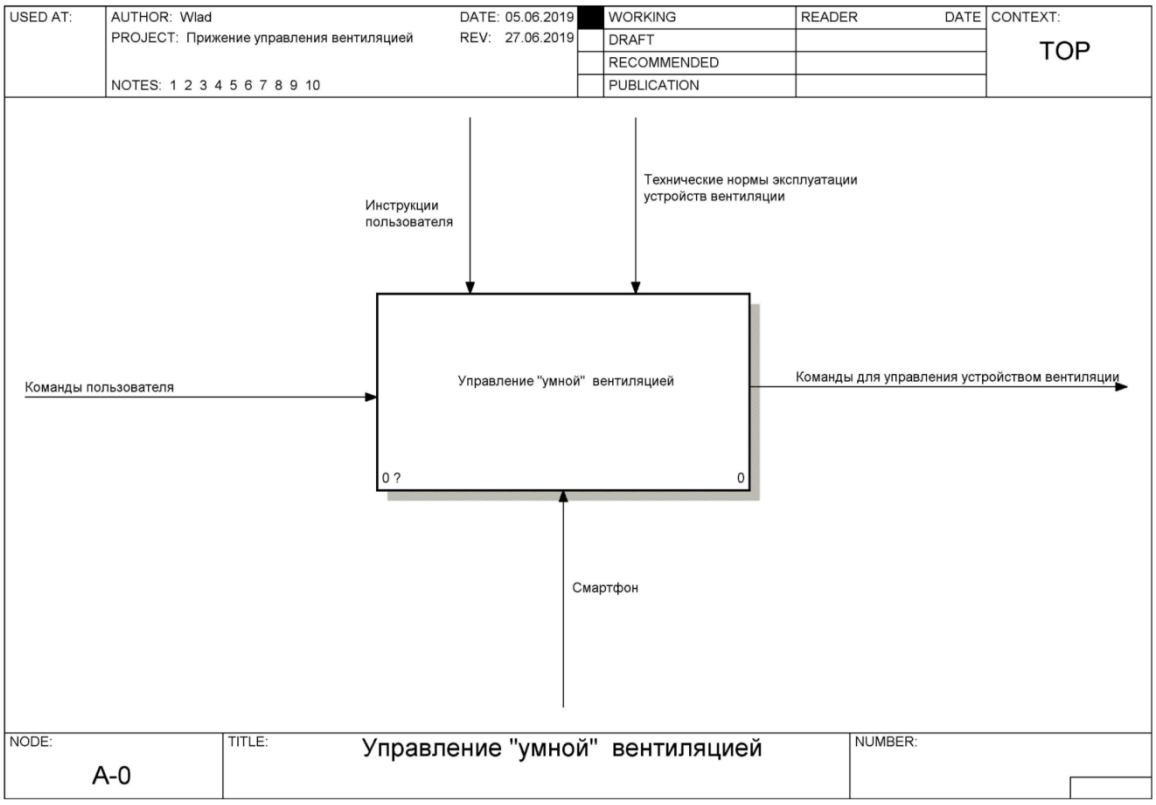


Рисунок 16 – Контекстная диаграмма «Устройство управления «умной» вентиляции» КАК БУДЕТ

Далее показана декомпозиции контекстной диаграммы.

Диаграмма показывает общую структуру логики работы приложения, основные модули приложения.

Работа приложения будет разделяться на следующие процессы:

- обработка команд пользователя;
- обработка данных с датчиков

– обработка исключительных ситуаций

– передача команд на устройство управления вентиляции.

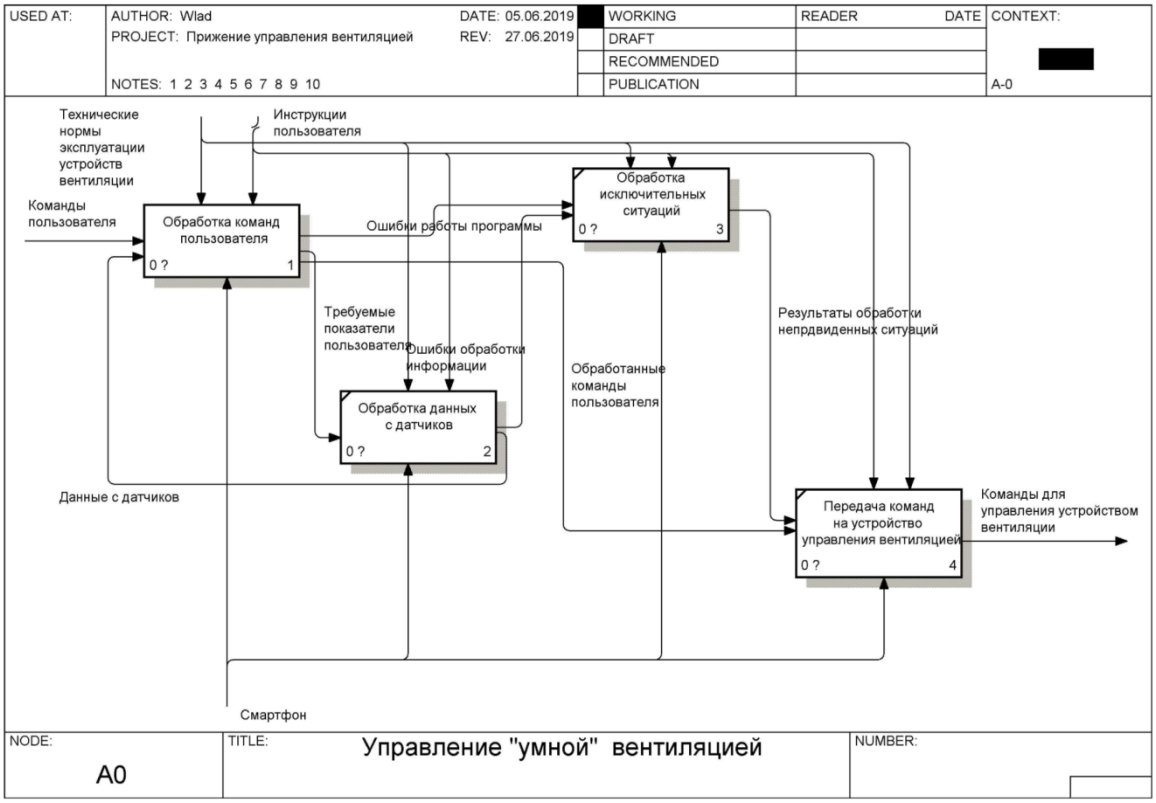


Рисунок 17 – Диаграмма декомпозиции «Устройство управления вентиляции»

Модуль «Обработка команд пользователя» будет принимать данные выбора в мобильном приложении, приводить информацию к необходимому шаблону и обрабатывать их.

Модуль «Обработка данных с датчиков» будет получать информацию с датчиков устройства и обрабатывать их. На основании полученных данных будет выстраиваться некоторые условия работы систем вентиляции, которые позволят достичь выбранных параметров.

Модуль «Обработка исключительных ситуаций» будет выполнять проверку работы системы, на корректность данных и ошибок работы пользователя

Модуль «Передача команд на устройство управления вентиляции» будет выполнять приведение к необходимому формату данных пользователя, а также результаты работы приложения.

На рисунке 18 показана диаграмма декомпозиции «Обработка команд пользователя».

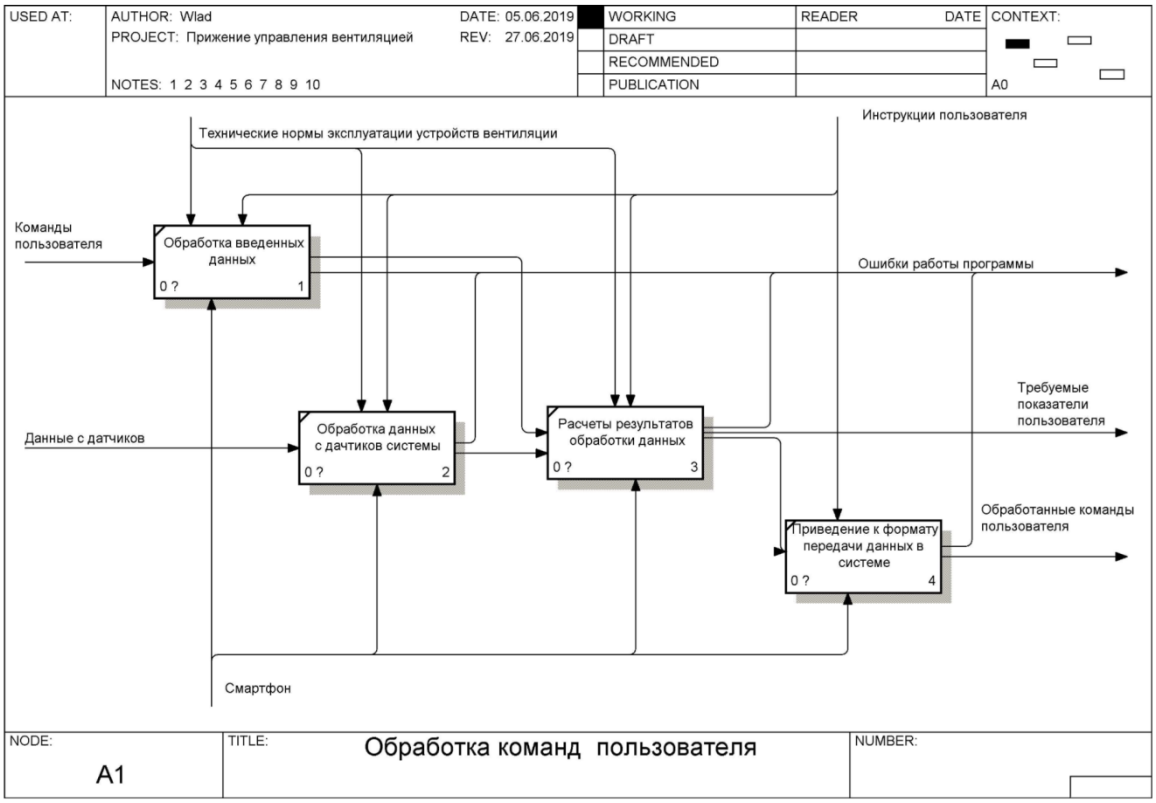


Рисунок 18 – Диаграмма декомпозиции «Обработка команд пользователя»

Таким образом, мобильное приложение будет обрабатывать значительную часть информации.

3.5 Разработка серверного приложения «HouseCloud»

Разработка данного приложения обусловлена тем, что обработку большинства данных необходимо вынести из расчетов на смартфоне в отдельное приложение, чтобы снизить нагрузку на процессор смартфона.

Серверное приложение будет написано на языке Node.JS. Платформа Node.js построена на базе JavaScript движка V8 от Google, который используется в браузере Google Chrome [7]. Данная платформа, в основном, используется для создания веб-серверов, однако сфера её применения этим не ограничивается.

Для работы сервера будет использована СУБД MySQL, которое является оптимальным решением для небольших и средних проектов [8]. Для работы приложения будет использоваться данная схема БД, показанная на рисунке 20.

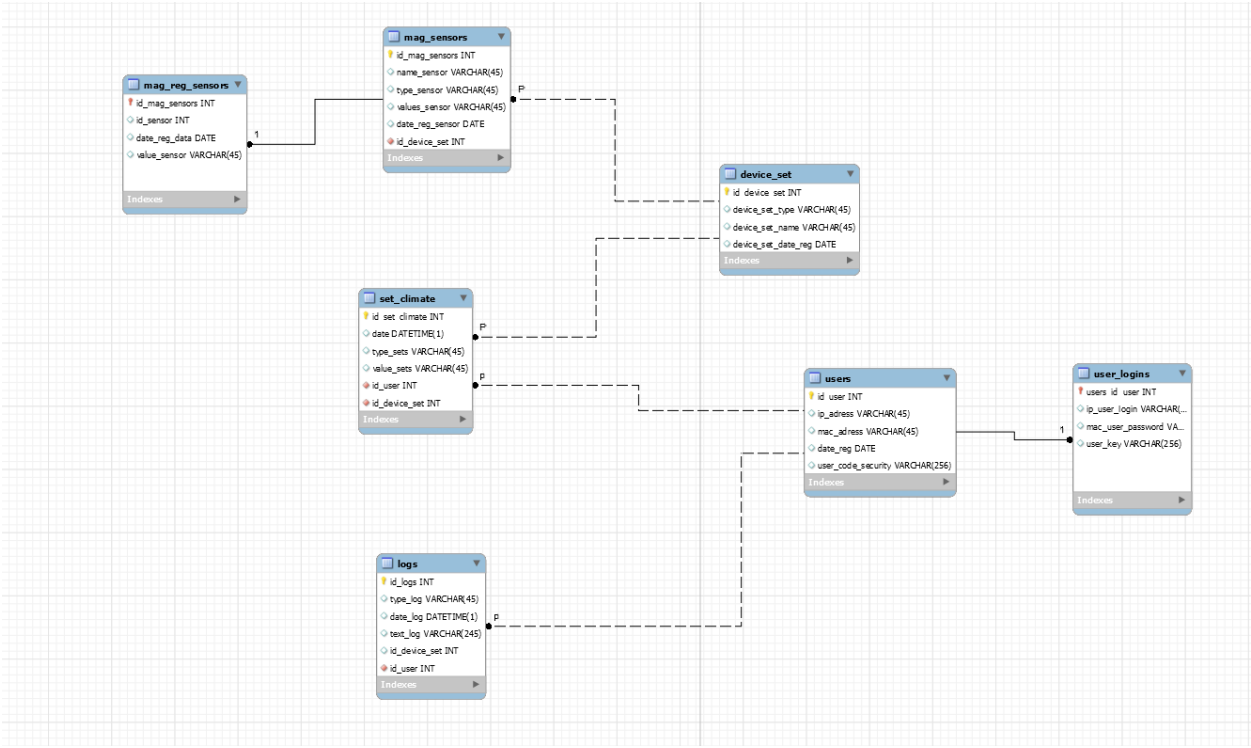


Рисунок 19 – Структура БД серверного приложения

Далее представлен код, который запускает работу серверного приложения.

```
require('dotenv').config();
const fs = require('fs');
const join = require('path').join;
const express = require('express');
const mongoose = require('mongoose');
const passport = require('passport');
const config = require('./config');
const models = join(__dirname, 'app/models');
const port = process.env.PORT || 3000;
const app = express();
```

Далее представлен код работы программы для обработки клиентских запросов:

```
(function () {
   angular.module('temp', [])
      .controller('tempController', function($scope) {
         var socket = io(); //
         //socket.emit();
         //console.log('aaa');
         $scope.locked = true; // Disable view by default

         socket.on('connect', function () { // On connection established
            $scope.locked = false; // Enable view
            $scope.$apply(); // Re-render view
         });
         socket.on('disconnect', function () { // Hide everything on disconnect
            $scope.locked = true;
            $scope.$apply();
         });
         socket.on('tps:state:changed', function (value1, value2) { // Catch 'tps:state:changed' event
            $scope.state = value == 0;
            $scope.$apply();
         });

         $scope.switch = function() { // Send inversed value of the 'state' variable
            console.log($scope.state ? 1 : 0);
            socket.emit('web:state:changed', $scope.state ? 1 : 0);
         }
      });
         })();
```

Данное приложение позволит получать HTTP запросы от мобильного приложения и устройства Arduino, которое будет контролировать «общение» между приложением и устройством. Именно серверное приложение позволит контролировать и управлять «общением» между устройством управления «умной» вентиляции и мобильным приложением. Тем самым, снимет с мобильного приложения большинство операций расчета и обработки информации, оставив ему лишь необходимую ему в этот момент.

Именно такой подход позволит решить поставленные задачи, поскольку также важна скорость работы приложения. И чтобы это реализовать, не нужно перегружать мобильное устройство различными вычислениями.

Также в работе серверного приложения будет организована обработка файлов формата JSON (JavaScript Object Notation – Нотация объектов JavaScript). Указанный формат позволит более продуктивно передавать данные по сети. Поддержка формата JSON встроена в JavaScript.

Далее представлен пример файла формата JSON, в котором описаны передаваемые данные на серверное приложение:


```
{
  "sensor_group_ID": 12345,
  "room_name": "Гостинная",
  "ip": "111.111.111.111",
  "mac_adress": "AA-AA-AA-AA-AA" [
    {
      "user_ID": 34,
      "user_name": "Ваня",
      "hash_function":
"381692e488413f5502fa7314a78c25db*48*e5bf81a2a23c88f3dccb44bc7da68bb5606b653b733bc
f9adaa5eb2c8ccf53abba66539044eb1957eda68469b1d0b9b5*48*b222df06deb308bf919d13447e6
88775fdcab972faed2c866dc023a126cb4cd4bbffab3683ecde243cf8d88967184680"
    },
    {
      "temp": 56,
      "humidity": "0,56",
    }
  ],
  "set_Completed": true
          }
```


3.6 Тестирование приложения «HouseCloud»

В данном разделе будет показана работа мобильного приложения, которое получает данные с устройства управления.

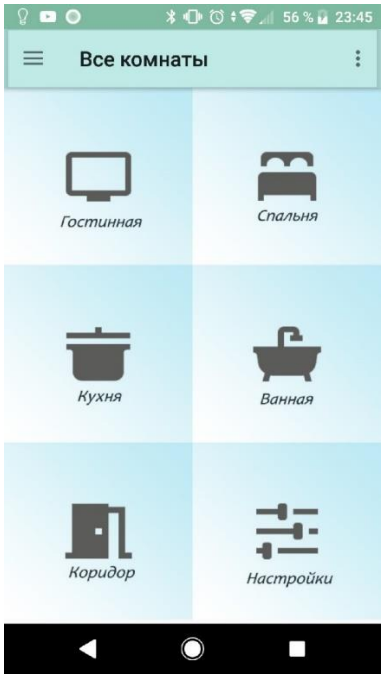


Рисунок 20 – Главное меню «HouseCloud»

При запуске приложения открывается главная страница, показанная на рисунке 20. В зависимости от сферы использования плитки кнопок могут быть изменены.

Рисунок 21 – Карточка комнаты «Спальня»

Данные формы управления универсальны для большинства помещений.

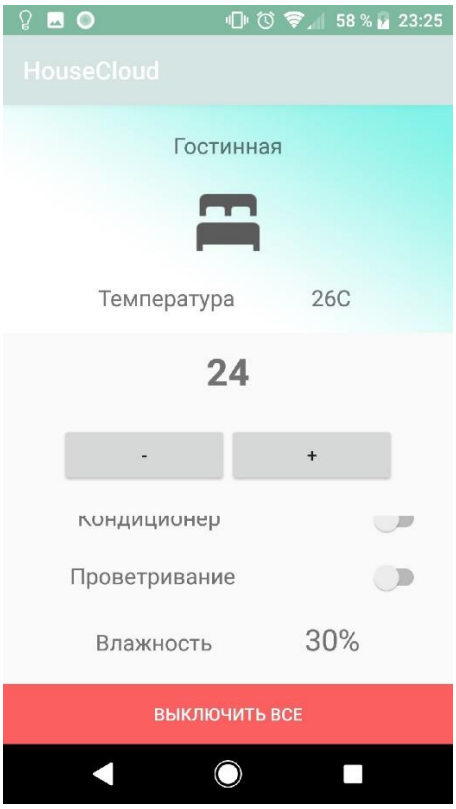


Рисунок 22 – Карточка комнаты «Гостиная»

Далее при выборе одного из элементов главного экрана, открывается форма управления параметрами климата в помещении. Например, на рисунке 22 показана температура и влажность в помещении, а также есть возможность выбора других функций управления климатом в помещении.

При использовании приложения, например, выставив температуру в гостиной 24 градуса Цельсия, сервер получил JSON-файл:

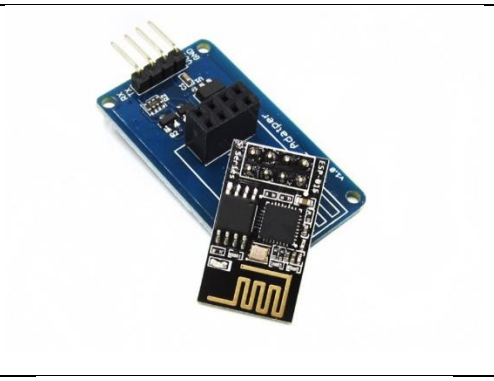
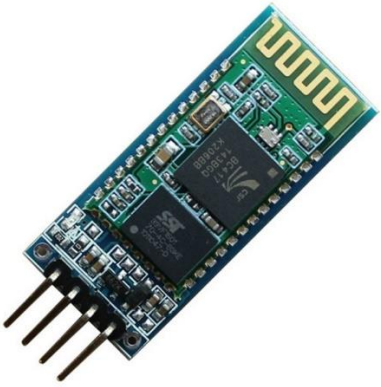

```
{
  "sensor_group_ID": 000001,
  "room_name": "Гостинная",
  "ip": "192.168.0.134",
  "mac_adress": "3C-5F-2D-4A-5F-9F" [
    {

      "user_ID": 01,
      "user_name": "Владислав",
      "hash_function":
"381692e488413f5502fa7314a78c25db*48*e5bf81a2a23c88f3dccb44bc7da68bb5606b653b733bc
f9adaa5eb2c8ccf53abba66539044eb1957eda68469b1d0b9b5*48*b222df06deb308bf919d13447e6
88775fdcab972faed2c866dc023a126cb4cd4bbffab3683ecde243cf8d88967184680"
    },

    {
      "temp": 24,
      "humidity": "0,67",
    }

  ],

  "set_Completed": true

          }
```


3.7 Стоимость компонентов разработки прототипа и расчет стоимости системы «умной» вентиляции «HouseCloud»

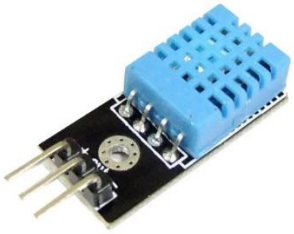
Каждое новое устройство необходимо оценить. Данный раздел направлен на экономическую оценку рентабельности системы на российском рынке.

Для начала приведем стоимость всех комплектующих для устройства в таблице 2.

Таблица 2 – Стоимость компонентов устройства

Наименование устройства	Изображение устройства	Стоимость, рублей
Arduino Nano		260
Модуль WiFi ESP8266-01		230
Модуль Bluetooth HC-06		330
Модуль ИК передатчика		140

Продолжение таблицы 2

Экран 1602 I2C		240
Датчик температуры и влажности DHT11		100

Исходя из приведенной выше таблицы, можно сделать вывод, что стоимость необходимых комплектующих невысока.

Далее рассмотрим стоимость разработки системы «HouseCloud».

Внедрение приложения на Play Market 2000 рублей.

Трудозатраты потраченные на проектирование системы составляют 2000 рублей.

$$S_{РП} = S_р + S_д; \quad (1)$$

где:

$S_{РП}$ – суммарные затраты на разработку системы «HouseCloud»(руб.);

$S_р$ – затраты на проектирование системы;

$S_д$ – прочие расходы на разработку.

$S_{РП} = 2000 + 1210 = 3\,210$.

Далее рассчитаем примерную розничную стоимость и примерную выручку за продажу 100 устройств «HouseCloud».

$$Z_П = S_{РП} + П; \quad (2)$$

где:

$Z_П$ – оптовая цена (цена разработчика) (руб.);

П – прибыль, рассчитанная по формуле:

$$П = \rho_н \cdot S_{РП}; \quad (3)$$

где:

$\rho_н$ – норматив рентабельности, учитывающий прибыль организации, разрабатывающей данную программу в долях ко всем затратам данной организации на разработку программы. Итак,

$$Z_П = S_{РП} \cdot (1+ \rho_н); \quad (4)$$

$Z_{П =}$ 3210*1,25 = 4012,5 рублей;

Затраты на производство 100 устройств составит

$B_о$ = 3210 * 100 + 2000= 323 000 рублей;

Это при учете, что продастся только 100 систем.

Розничная цена программы рассчитывается с учетом налога на добавленную стоимость (НДС = 20%) по формуле:

$$Z_{ПР} = Z_П + НДС; \quad (5)$$

$Z_{ПР}$ = 4012,5 * 1,2 = 4 815 рублей;

Выручка от продаж составит:

$$B_р = Z_{ПР} \cdot n_п; \quad (5)$$

где:

$n_п$– количество организаций, желающих прибрести устройство соответствует количеству 100.

$B_р = 4815 * 100 = 481\ 500$;

Если сравнивать с уже имеющимися устройствами управления вентиляцией, то сумма 4815 рублей не сравниться, против 8 000 рублей.

Аналогов в данном ценовом сегменте нет.

Вывод по разделу 3

Выводы по третьему разделу

В данном разделе была спроектирована система «умного» управления вентиляцией, в которую включены следующие элементы: прототип устройства управления вентиляцией, программа для этого устройства, мобильное приложение, серверное приложение «HouseCloud». Также представлены актуальные цены на компоненты и расчет стоимости системы.

ЗАКЛЮЧЕНИЕ

В выпускной квалификационной работе был показан процесс совершенствования управления устройствами вентиляции за счет разработки прототипа системы «умной» вентиляции на платформе Arduino. В данную систему предположительно включены устройства: кондиционеры, системы увлажнения воздуха, ионизаторы и другие устройства. Также для этой цели были разработаны мобильное и серверное приложения «HouseCloud».

В процессе создания мобильного и серверного приложений использовались следующие технологии и инструменты: Java, XAML, NodeJS, SQL, Adobe Photoshop, AndroidStudio, SublimeText 3, MySQL Workbench.

Несомненно, в наше время такие системы актуальны. Можно сказать точно, что они будут развиваться и совершенствоваться дальше для увеличения производительности, удобства, а также качества обслуживания клиентов.

Для совершенствования систем «умной» вентиляции можно выделить следующие проекты:

- разработка мобильного приложения для платформы IOS;
- создание графического интерфейса серверного приложения;
- увеличение количества поддерживаемых управляемых устройств.

Подводя итог проведенной работе можно сказать, что поставленные задачи выполнены в полном объеме, таким образом цель работы можно считать достигнутой.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. А.Беккер. Системы вентиляции. – М.: Техносфера, 2017. – 252 с.
2. У.Соммер. Программирование микроконтроллерных плат Arduino/Freeduino. – СПб.: БХВ-Петербург, 2014. – 256 с.
3. Т.Иго. Arduino, датчики и сети для связи устройств. – СПб.: БХВ-Петербург, 2015. – 544 с.
4. Android. Программирование для профессионалов. 2-е изд. – СПб.: Питер, 2016. – 640 с.: ил. – (Серия «Для профессионалов»). ISBN 978-5-496-02051-0.
5. AllFusion Process Modeler r7 (BPwin). Описание AllFusinon Process Modeler r7 (BPwin) [электронный ресурс] / ITShop интернет-магазин –Режим доступа: http:/www.interface.ru/home.asp?artId=106.
6. Документация по Node.JS [Электронный ресурс]. Режим доступа https://nodeguide.ru/doc/.
7. Документация по MySQL [Электронный ресурс]. Режим доступа http://www.sql.ru/docs/mysql/rus_ref/.
8. Android для разработчиков. 3-е изд. – СПб.: Питер, 2016. – 512 с.: ил. – (Серия «Библиотека программиста»). ISBN 978-5-496-02371-9
9. 9-е полное издание. – М.: Вильямс, 2015. – 1377 с. – ISBN 978-5-8459-1918-2. Книга Java 8.
10. 2-е изд. – М.: Эксмо, 2012. – 708 с. – ISBN 5699545743, 9785699545742. Изучаем Java.
11. BPwin и Erwin. CASE-средства для разработки информационных систем (fb2) [Электронный ресурс]. Режим доступа: http://coollib.com/b/147153/read.
12. Проектирование информационных систем. [Электронный ресурс]. Режим доступа: https://damirock.com/hse/pris3.

13. Информационная безопасность систем организационного управления. Теоретические основы. В 2 томах. Том 1: моногр. . - М.: Наука, 2018. - 496 с.

14. Калашян, А.Н. Структурные модели бизнеса: DFD-технологии / А.Н. Калашян. - М.: Финансы и статистика, 2017. - 526 с.

15. Москвин, Э. К. Локальная сеть без проводов / Э.К. Москвин. - М.: НТ Пресс, 2016. - 128 с.

16. Филиппов, В. А. Электронные хранилища информации и WEB-технологии / В.А. Филиппов. - М.: Едиториал УРСС, 2015. - 588 с.

17. Конструируем роботов на Arduino. Умный свет / А.А. Салахова. – М. :Лаборатория знаний, 2017. – 48с.

18. Конструируем роботов на Arduino, Первые шаги / Дж. Бейктал; пер. с англ. О.А. Трефиловой. – М. : Лаборатория знаний, 2016. – 320 с.

19. Харди Б.,Филлипс Б., Стюарт К., Марсикано К. Android. Программирование для профессионалов. 2-е изд. – СПб.: Питер, 2016. – 640 с.: ил. – (Серия «Для профессионалов»). ISBN 978-5-496-02051-0.

20. Медникс З., Дорнин Л., Мик Б., Накамура М. П78 Программирование под Android. 2-е изд. — СПб.: Питер, 2013. — 560 с.: ил. — (Серия «Бестселлеры O’Reilly»). ISBN 978-5-496-00526-5.

21. Аудит безопасности информационных систем. — СПб.: Питер, 2018. — 272 с.: ил. — (Серия «Библиотека программиста»). ISBN 978-5-4461-0662-2.

22. Флэнаган Д. JavaScript. Подробное руководство, 6-е издание. – Пер. с англ. – СПб: Символ Плюс, 2012. – 1080 с., ил. ISBN 978 5 93286 215 5

23. Информационные системы: учебник для студ. Учреждений высш. образования / С.А. Жданов, М.Л. Соболева, А.С. Алфимова. – М.: ООО «Прометей» 2015. – 302с.

24. Федорова Г. Н. Информационные системы: учебник для студ. учреждений сред. проф. образования / Г. Н. Федорова. — 3-е изд., стер. — М. : Издательский центр «Академия», 2013. — 208 с.

Приложение А

Листинг мобильного приложения «HouseCloud»

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".gostin">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical"
        android:weightSum="14"
        tools:layout_editor_absoluteX="-16dp"
        tools:layout_editor_absoluteY="16dp">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_weight="3"
            android:background="@drawable/wall2"
            android:orientation="horizontal">

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:orientation="vertical"
                android:weightSum="9">

                <TextView
                    android:id="@+id/textView6"
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
                    android:layout_weight="4"
                    android:gravity="center_horizontal"
                    android:paddingTop="20dp"
                    android:text="@string/gostin"
                    android:textSize="18sp" />

                <LinearLayout
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:orientation="horizontal">

                    <ImageView
                        android:id="@+id/image_gostin"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content"
                        android:layout_marginTop="20dp"
                        android:layout_marginBottom="20dp"
                        android:layout_weight="2"
                        app:srcCompat="@drawable/spaln2" />
                </LinearLayout>

                <LinearLayout
                    android:layout_width="match_parent"
                    android:layout_height="match_parent"
```

```
            android:layout_weight="4"
            android:gravity="center|fill_horizontal|center_vertical|fill_vertical"
            android:orientation="horizontal"
            android:weightSum="5">

            <TextView
                android:id="@+id/textView2"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="3"
                android:gravity="center_horizontal"
                android:paddingLeft="15dp"
                android:text="Температура"
                android:textSize="18sp" />

            <TextView
                android:id="@+id/text_temp_now_gostin"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="2"
                android:text="26C"
                android:textSize="18sp" />

            <android.support.constraint.Guideline
                android:id="@+id/guideline"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:orientation="horizontal" />
        </LinearLayout>

    </LinearLayout>

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_weight="4"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/text_temp_set_gostin"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:layout_weight="1"
        android:gravity="center"
        android:text="24"
        android:textSize="30sp"
        android:textStyle="bold"
        android:verticalScrollbarPosition="left" />
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_weight="4"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:layout_weight="1" />

    <Button
        android:id="@+id/button_minus_gostin"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="-" />

    <Button
        android:id="@+id/button_plus_gostin"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="+" />

    <TextView
        android:id="@+id/textView5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1" />
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_weight="3"
    android:orientation="vertical"
    android:weightSum="10">

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:scrollbarStyle="insideOverlay">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:layout_marginTop="20dp"
                android:layout_weight="1"
                android:fadingEdge="horizontal|vertical"
                android:orientation="horizontal"
                android:weightSum="3">

                <TextView
                    android:id="@+id/text_auto_gostin"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_weight="2"
                    android:gravity="center"
                    android:text="@string/auto_set"
                    android:textSize="18sp" />

                <Switch
                    android:id="@+id/switch_auto_gostin"
                    android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:layout_marginRight="20dp"
        android:layout_weight="1" />
    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginTop="20dp"
        android:layout_weight="1"
        android:orientation="horizontal"
        android:weightSum="3">

        <TextView
            android:id="@+id/textView9"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="2"
            android:gravity="center_horizontal"
            android:text="@string/kondi"
            android:textSize="18sp" />

        <Switch
            android:id="@+id/switch_kondi_gostin"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginRight="20dp"
            android:layout_weight="1" />

    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginTop="20dp"
        android:layout_weight="1"
        android:orientation="horizontal"
        android:weightSum="3">

        <TextView
            android:id="@+id/text_provetr_gostin"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_weight="2"
            android:gravity="center_horizontal"
            android:text="@string/provetr"
            android:textSize="18sp" />

        <Switch
            android:id="@+id/switch_provetr_gostin"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginRight="20dp"
            android:layout_weight="1"
            android:checked="false"
            tools:checked="false" />

    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginTop="20dp"
```

```
                android:layout_weight="2"
                android:orientation="horizontal"
                android:weightSum="3">

                <TextView
                    android:id="@+id/text_vlazhn_t_gostin"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_weight="2"
                    android:gravity="center_horizontal"
                    android:text="@string/vlazhn"
                    android:textSize="18sp" />

                <TextView
                    android:id="@+id/text_vlazhn_gostin"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:layout_weight="1"
                    android:text="30%"
                    android:textSize="24sp" />
            </LinearLayout>
            <Button
                android:id="@+id/button_off_all_gostin"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginTop="20dp"
                android:layout_weight="2"
                android:background="@color/red_button"
                android:text="Выключить все"
                android:textColor="@android:color/background_light" />
        </LinearLayout>          </ScrollView>
    </LinearLayout>
  </LinearLayout>
</android.support.constraint.ConstraintLayout>
```